

Grid Programming Methods
University of Tampere, 2nd Feb 2006
Marko Niinimäki
man@sjoki.uta.fi

Summary: Grid programming \approx
using Semantic Web methodologies
with PKI
authentication+authorization.
Special emphasis in large data and
time consuming distributed
computing.

Sources with influence:
Foster & Kesselman: The Grid:
Blueprint for a New Computing
Infrastructure, 1998, 2nd ed 2003.
Li & Baker: The Grid – Core
Technologies, 2005.

Contents:
The Grid
Authentication & Authorization
Programming interfaces
Applications

Comments welcome! Especially:

- Are there similar/
better/easier/more flexible
technologies available?
- Can you think of an
application?



What is the Grid?



A Grid is a system which

- coordinates resources that are not subject to centralized control (A Grid integrates and coordinates resources and users that live within different control domains)
- using standard, open, general-purpose protocols and interfaces (for authentication, authorization, resource discovery, and resource access)
- to deliver nontrivial qualities of service.

[Foster: What is the Grid, a three point checklist, GridToday July 2002]

What are Grid resources and how are they coordinated?



“From a hardware perspective a Grid is a collection of distributed resources connected by a network, possibly at different sites and in different organizations.

Those resources may include terascale supercomputers, instruments such as telescopes and microscopes, computer-controlled factory floor tools, mid-level servers, desktop machines, laptops, PDA's, and [...] video cameras, cell phones, and kitchen appliances.”

It is a task of a **Grid middleware** to virtualize, and coordinate the access and use to these resources.

Grimshaw: What is a Grid? GridToday, Dec 2002.

How about protocols and “non-trivial qualities of service”?



Pro
to
col
sta
cks

GridFTP -- Application

GSI (Grid Security Infrastructure) -- AA
--

SSL/TSL (Secure Sockets Layer/Transport Security Layer)

Web Service applications

gSoap

SSL/TSL -- OpenSSL

QoS: an error tolerant distributed storage and computing system.

Grid Middlewares (“Grid systems”)



3 generations (Catlett: The rise of Third Generation Grids, GridToday Oct 2003):

1G Grids: local "Metacomputers" with basic services such as distributed file systems and site-wide single sign on.

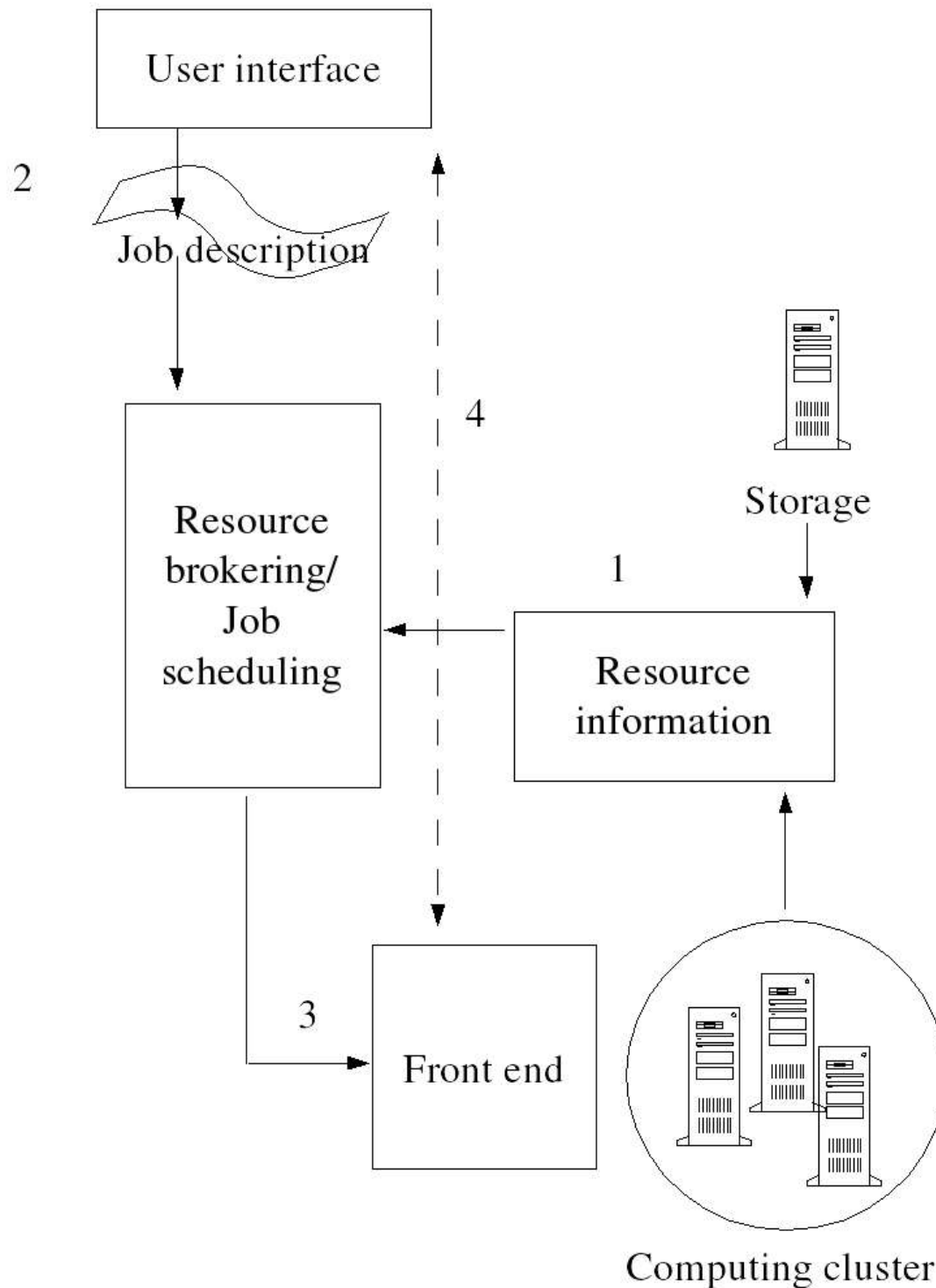
2G Grids: underlying software services and communications protocols could be used as a basis for developing distributed applications and services. 2G systems were/are mutually incompatible.

3G architectures, Open Grid Services Architecture (OGSA), a set of common interface specifications supports the **interoperability of discrete, independently developed services.**

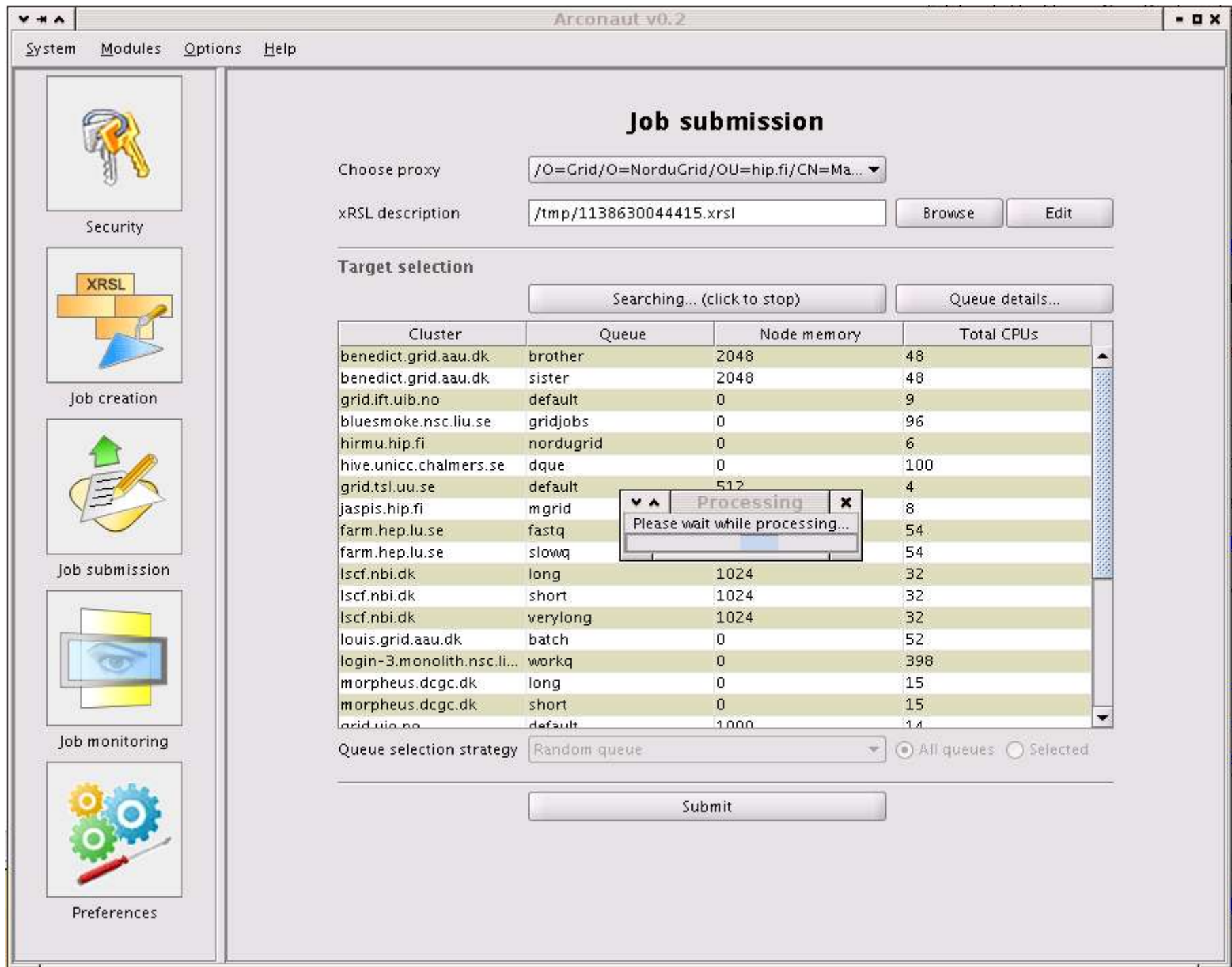
Grid



Components of a typical (2G) Grid



Niinimäki, Toivonen, Niemi:
Modelling, searching and
utilising Grid resources, EJC
2005.



A GUI for NorduGrid's ARC middleware (Arconaut by Ilja Livenson)

Describing the interfaces



Programming view: a Grid service is a collection of “Grid methods”

A method is

- declared using Web Service Description Language
- called by its name and parameters, but the call is wrapped inside a Transport Security Layer/Secure Sockets Layer that uses X.509 certificates for mutual authentication between the service and the caller (actual data transport is SOAP)

Example: method “hi” using GridSite and a simple Perl/CGI.



The server side

```
use SOAP::Transport::HTTP;

SOAP::Transport::HTTP::CGI
    -> dispatch_to('Demo')

    -> handle;

package Demo;

=begin WSDL

    _IN foo $string The string to be echoed.

    _RETURN $string The string echoed with hello prepended.

=end WSDL

sub hi {

    $class = shift;

    $a = shift;

    return "hello $a";

}

1;
```



Example: continued

Generating WSDL and the resulting WSDL file hi.wsdl

```
use Pod::WSDL;

my $pod = new Pod::WSDL('source' => '/var/www/htdocs/cgi-bin/hi.pl',
location => 'https://tuovidev.cern.ch:8443/gs-cgi-bin/hi.pl',
pretty => 1,
withDocumentation => 1
);

print $pod->WSDL;

<wsdl:definitions targetNamespace="https://tuovidev.cern.ch:8443/Demo"..

    <wsdl:message name="hiRequest">

        <wsdl:part name="foo" type="xsd:string">

            <wsdl:documentation>The string to be echoed.</wsdl:documentation>

        </wsdl:part>

    </wsdl:message>

...

```



Example: continued

Calling the service (a simple client program client.pl)

```
use SOAP::Lite; $soaplite = SOAP::Lite -> service('http://pcephc23.cern.ch:8080/Grid.wsdl');
```

```
$name = <STDIN>; $n = $soaplite->hi($name); print '$n\n';
```

```
perl client.pl
```

```
man
```

```
hello man
```

By Java/Javascript somewhat similar but cannot paste all here..

```
function hi(v1) { .. response = proxy(hi); .. }
```

```
function createProxy(aCreationListener) { var factory = new WebServiceProxyFactory(); factory.createProxyAsync(wSDL_uri, "myService", "", true, aCreationListener); .. }
```

```
function Hello (aValue1) { if (!proxy) { var listener = {  
    onLoad: function (aProxy)  
    { proxy = aProxy; proxy.setListener(listener); hi(aValue1); },  
    ..
```

What about authentication/authorization?

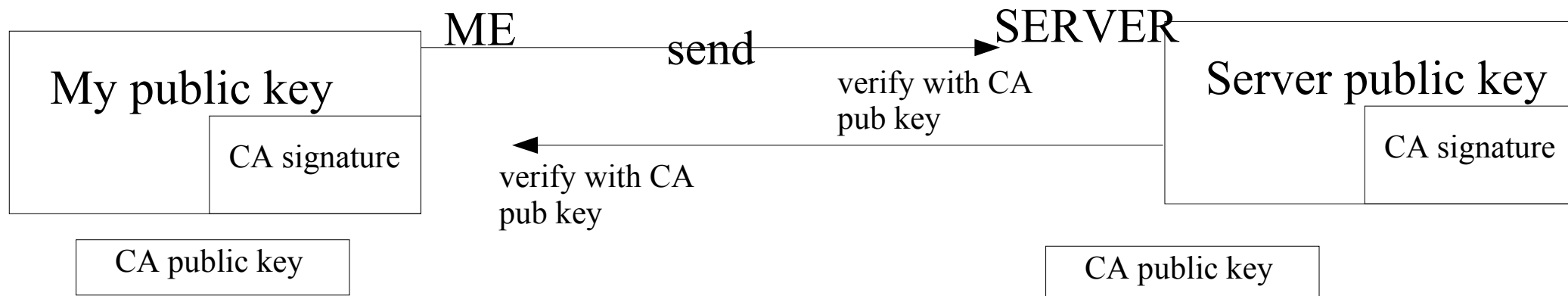


- Managed by security libraries (typically Globus or openssl) at both the client and server ends.
- Grid Security Infrastructure (GSI) uses a specific proxy file for delegating the user's authority to the server (generally: middleware, GSI-enabled services). The proxy file can contain authorization related extension (e.g. “voms”).
- For the programmer, this is mainly invisible (not too many additions in the code).

Authentication/authorization with certificates



- X.509 certificates are practically digitally signed forms of identity with predefined field names.
- An entity (person or computer) carries two files, a public key and a private key. The public key is signed by a Certification Authority.
- The following figure is too simplified but gives the basic idea of **mutual authentication**:



Authentication and authorization (AA) in the example



Adding “Grid” AA to the software setup:

- Server side: configure Apache/GridSite to use server certificates. Free bonus: “.gacl” files that control the access to the server's directories by certificate subjects.
- Client side by browser: import your certificate in the browser.
- Client side standalone example program:

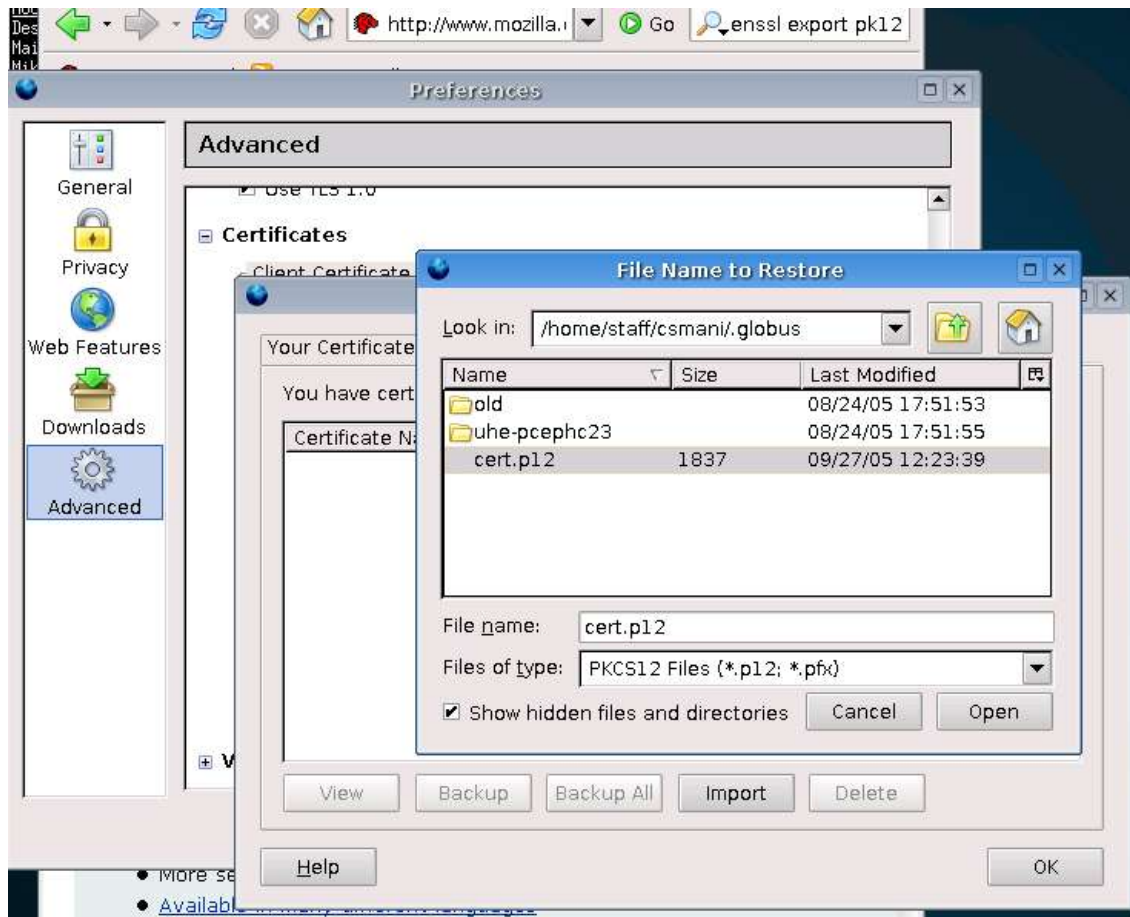
```
use Crypt::SSLey;
```



AA in the example, cont'd

Importing the certificate

Directing the browser to a simple JavaScript page that provides a form and executes “hi” with the parameter given by the user



hello

Enter name:

hello man,



So?

- The previous example illustrated one client connecting to one server. Production Grids have hundreds (or thousands) of computers and services.
- Examples of relatively interesting Grid programming tasks (built on top of previous technologies): distributed resource/service/data directories, brokers, instrument interfaces, large middleware projects (Globus, EGEE, UNICORE, NorduGrid ARC).

Grid Monitor

2006-01-25 CET 09:29:24



Processes: ■ Grid ■ Local

Country	Site	CPUs	Load (processes: Grid+local)	Queue
Denmark	Benedict - Aalborg pr>	48	46+0	5+0
	Horseshoe (DCSC/SDU)	1159	0+821 (queue down)	558-
	Louis XIV (DCGC/AAU)	52	28+1	0+0
	Morpheus	16	2+0	0+0
Estonia	Tartu Observatory	5	0+0	0+0
	TUIT DOUG cluster	1	0+0	0+0
	UT CS Antarctica Clus>	19	0+0	0+0
	UT DP/ITech/IPhys Ear>	12	0+0	0+0
	UT IMCB Anakonda clus>	13	0+0	0+0
	UT Physics Cluster	0	0+0	0+0
Finland	Akaatti (M-grid)	30	0+26	0+5
	Ametisti (M-grid)	132	0+59	0+5
	Hirmu Cluster (HIP)	6	0+0	0+0
	Jaspis (M-Grid, HIP)	8	0+0	0+0
	Kvartsi (M-grid)	96	0+78	0+0
	Opaali (M-grid)	24	0+16	0+0
	Spektrolitti (M-grid)	26	0+24	0+0
	Topaasi (M-grid)	24	0+16	0+0
Germany	LRZ cluster	296	0+246	0+4
Lithuania	grid.ktu.lt	13	1+0	0+0
Norway	Bergen Grid Cluster	11	0+3	1+0
	IBM 1300 cluster - Fi>	38	0+7 (queue down)	3+0
	Norgrid @ NTNU	63	0+63	26+
	Oslo Grid Cluster	14	14+0	2+0

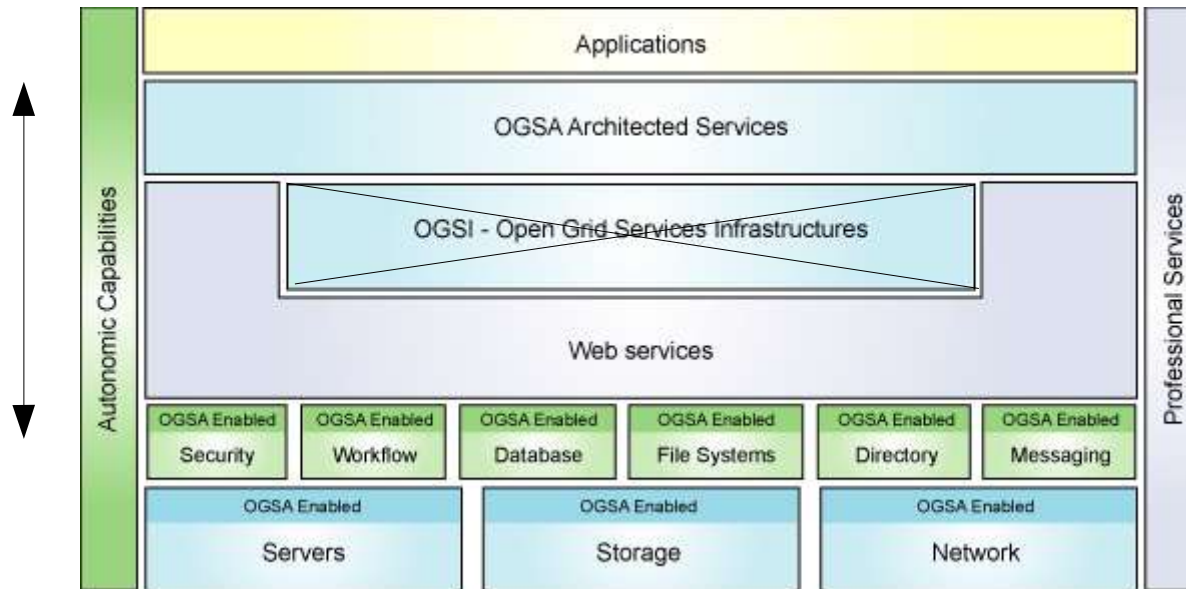
Done

NorduGrid: At the time of checking 50 sites, ~3800 CPU's.

Grid programming examples



Scope of
middleware?



3: See
following slides

2: Covered,
OGSI now
obsolete

1: See following
slides

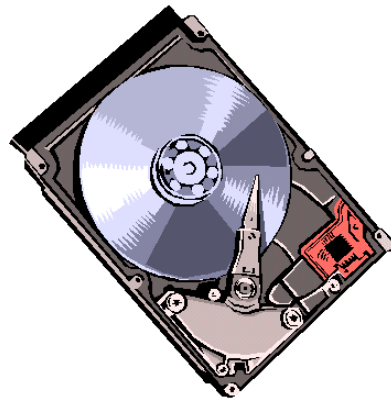
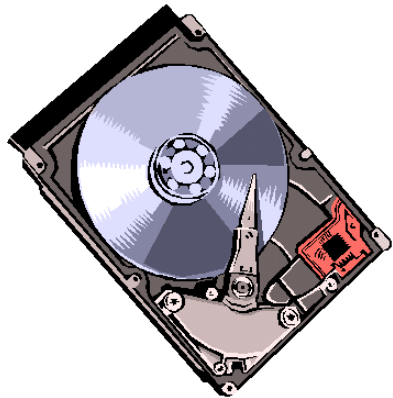
Unger & Haynos : A visual tour of Open Grid
Services Architecture, IBM, 2005.

Programming example 1



Enabling Grid storage interfaces.

Controlled access by several
protocols that use GSI.
Alternatives: gsiftp, https with
certificates.



A flexible “cheap hardware” disk
server at the department
gridstorage.cs.uta.fi (tricks: LVM,
ext2resize)

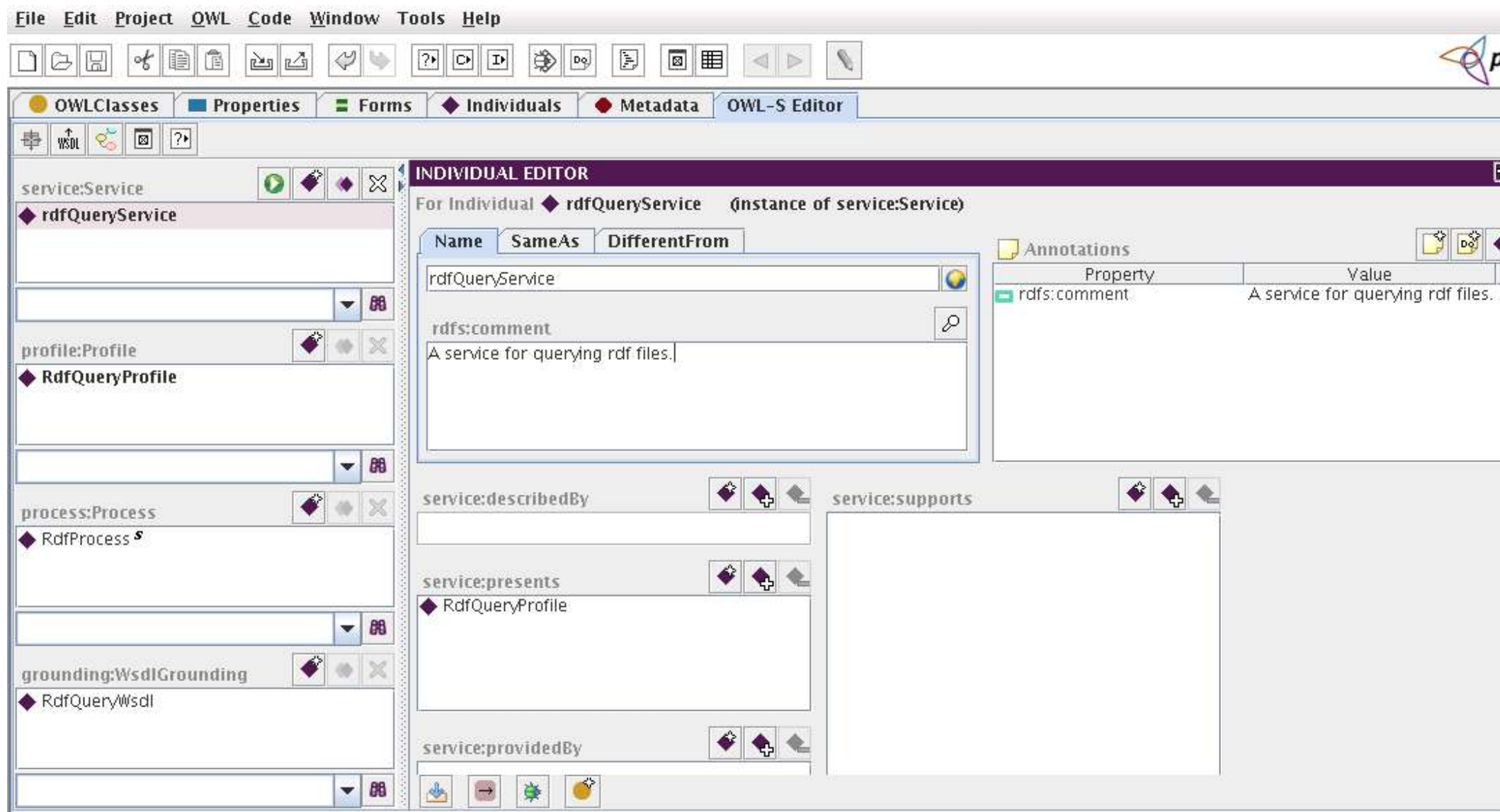
Programming example 3



An ontology based resource/data/service framework with a GUI <http://www.cs.uta.fi/reports/dsarja/D-2006-1.pdf>

The idea: Services, resources and data are described using RDF as in Semantic Web technologies. Concerning data: a structured description of the contents (XML data – tags, databases – table structures). Pointers to these descriptions stored in a data catalog. Will enable ad hoc queries like “for each data that you know, compute the trade value in dollars between European and Asian countries, sort by year”.

An application: dynamic OLAP cube construction.



An “ontology editor” for a service.

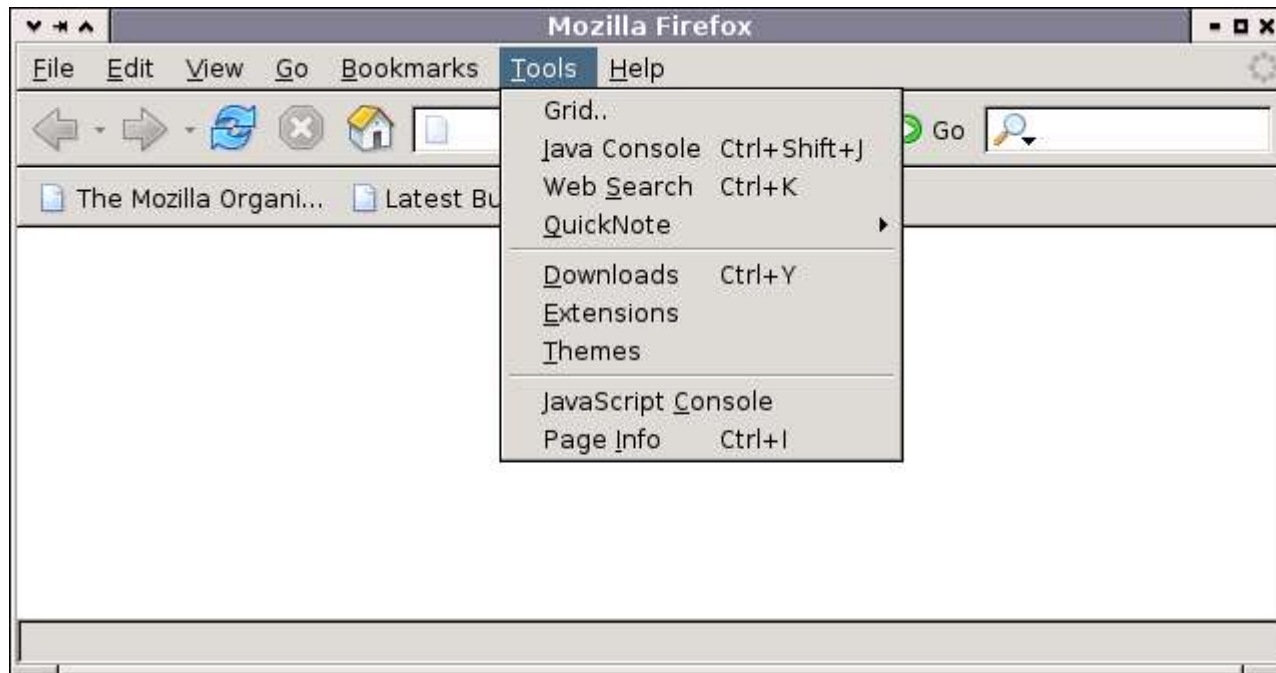


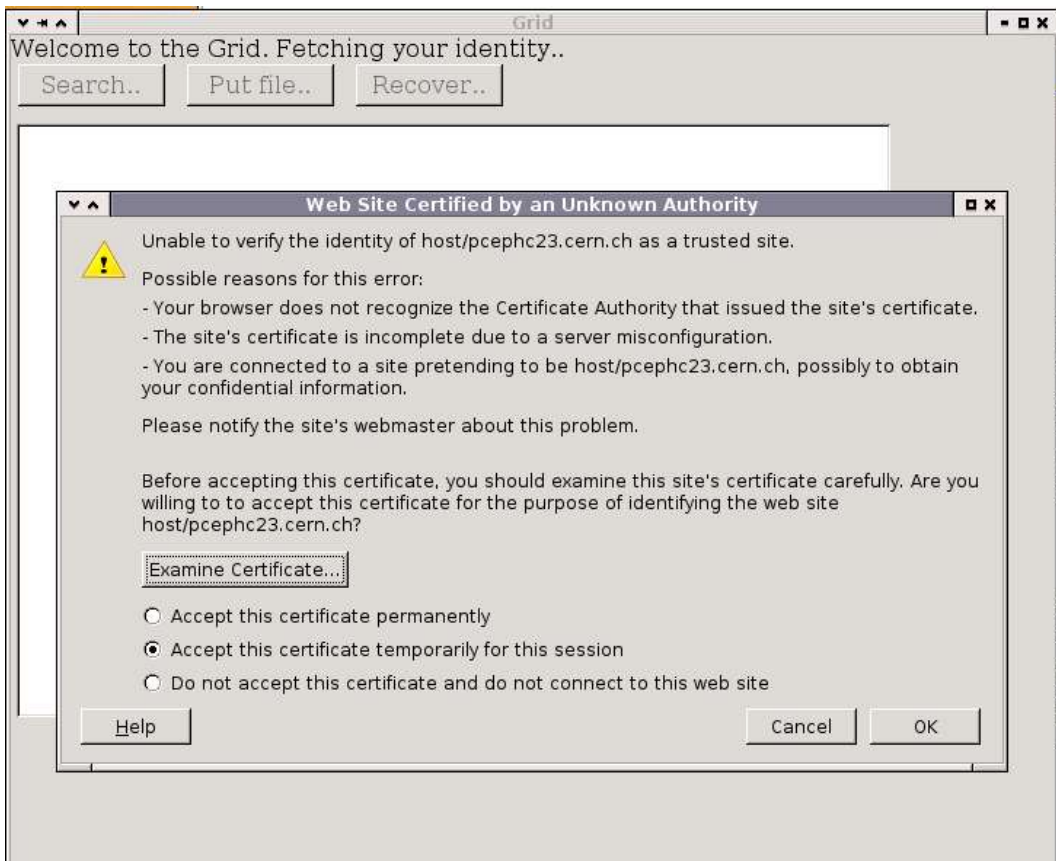
Programming example 3

Current implementation:

Server: as in the previous example, the cluster runs Condor.

Client: a Mozilla extension, “Grid in your browser”.





=NorduGrid/OU=hip.fi/CN=Marko Niinimaki

Put file.. Recover.. Find and run a service..

Other ongoing projects in Finland:

- Shibboleth/Grid: integrating identity management in various domains (HAKA infrastructure, Grid..)

In Helsinki Institute of Physics:

- Intrusion detection in Grid domains.
- Using certificate attributes/extensions for short term “ticket” certificates: “one can use this ticket for a limited time to assign 5 Euros from account X to account Y”. Related to Liberty Alliance/Grid integration.

Reference books for browsing, pointers to papers etc available at my office B1035.

Walkthrough of setting up Apache, GridSite, WSDL generation, clients:
<https://wiki.hip.fi/twiki/bin/view/MarkoPublic/SecureSoap>